Filters

7 min

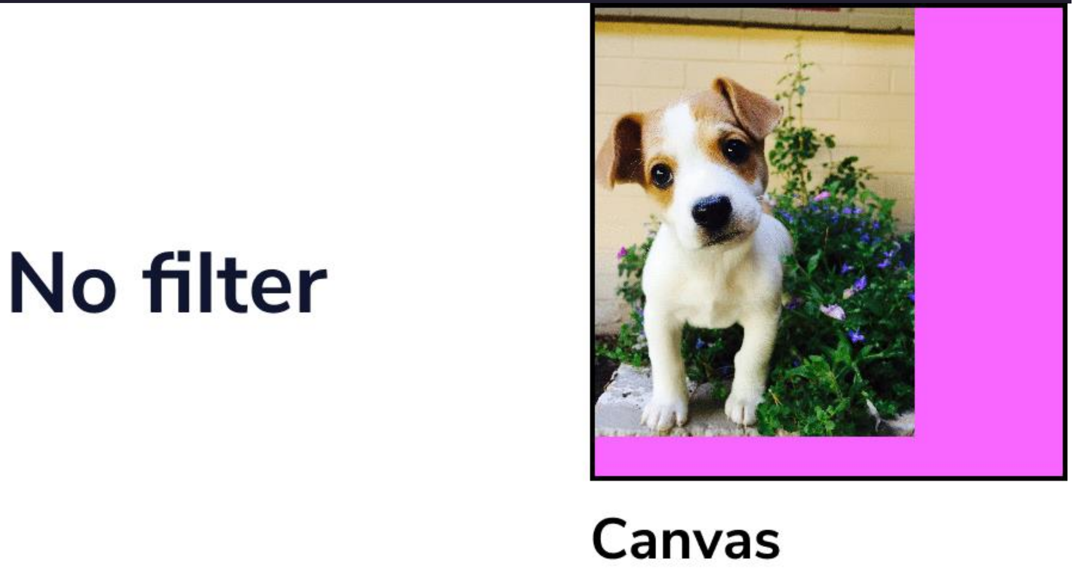
Adding photos and videos into the canvas is cool, but the real fun comes when we add effects to them through tools like filters.

Filters in p5.js work in two ways: across the entire canvas and applied to individual images.

To apply a filter across the canvas, use the `filter()` function. It requires the type of filter, which can be one of eight different types as defined by p5.js. For a complete list, visit the p5.js reference on filters.

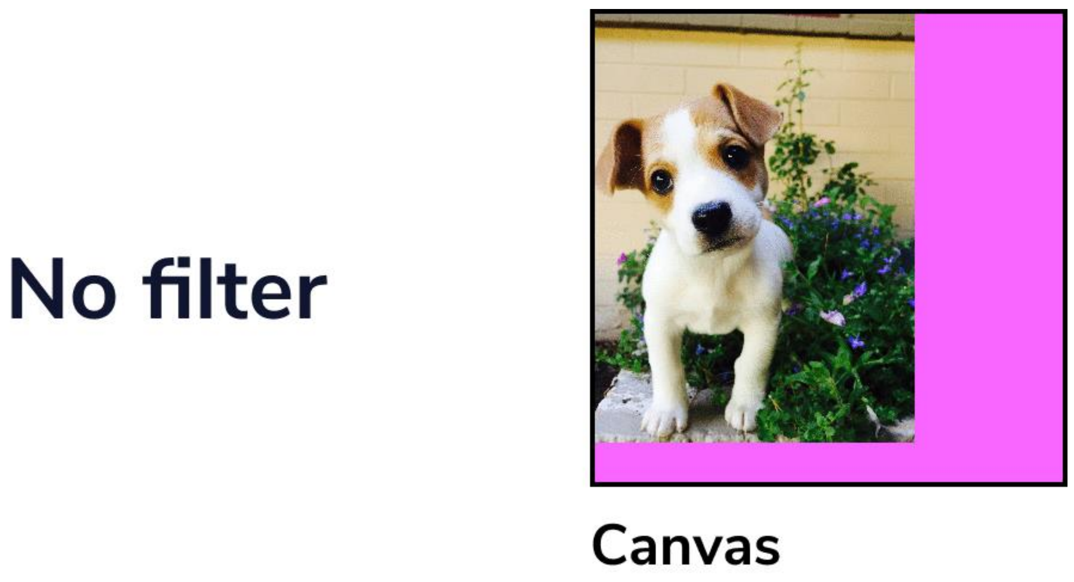
For example, to add a `GRAY` filter across the canvas, use:

```
filter(GRAY);
```

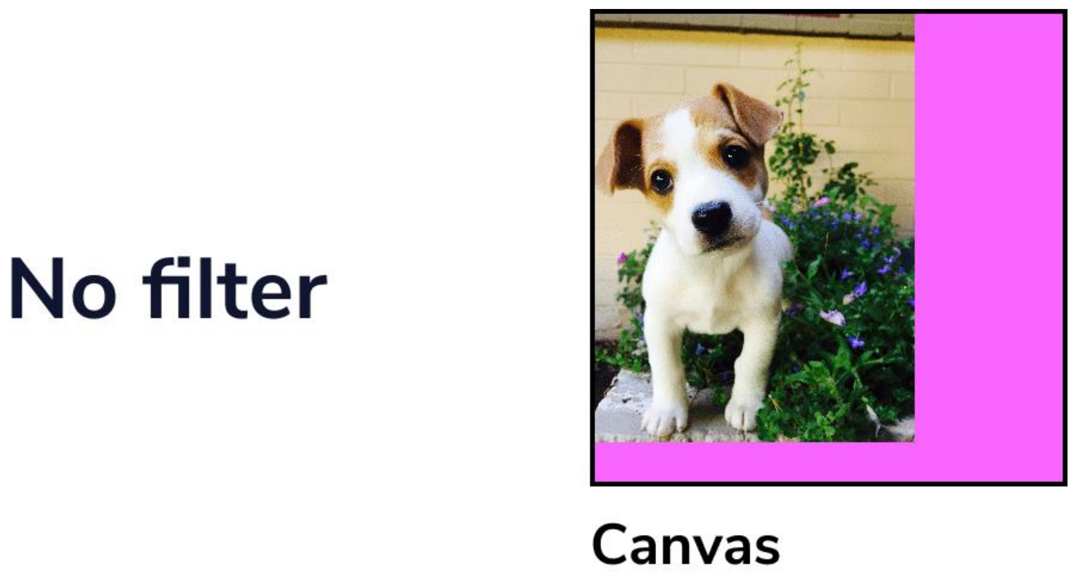


Certain filter types require an additional numerical argument. For example, the `POSTERIZE` filter, which reduces the number of colors in the image, requires a value between 2 and 255.

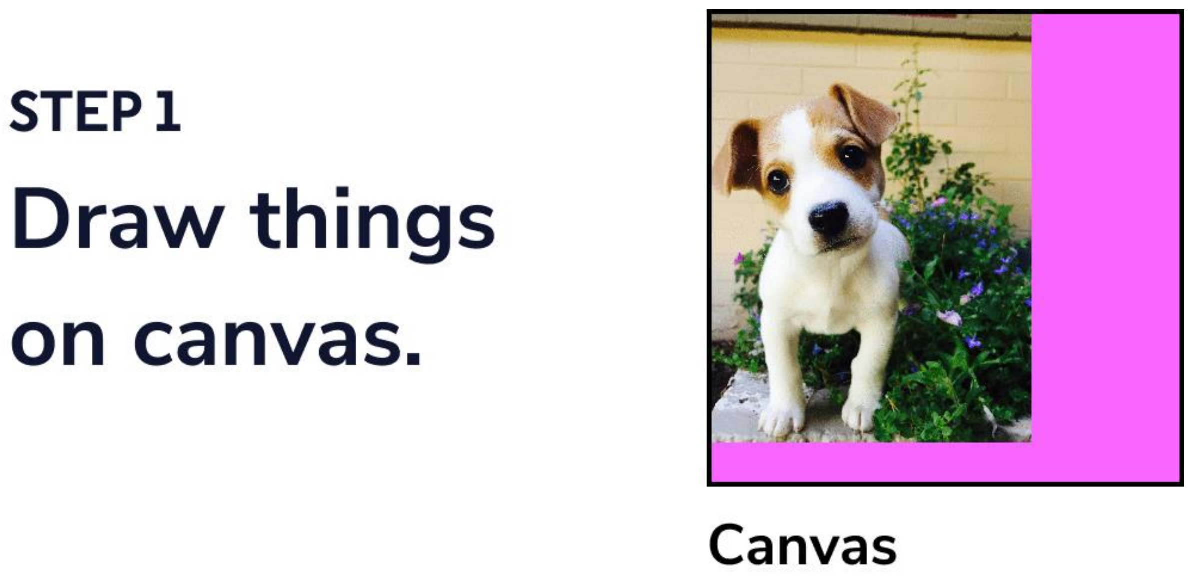
```
filter(POSTERIZE, 3);
```



The `filter()` function applies the filter to everything drawn on the canvas before it’s called—this lets us layer filters together, combining them with ones that were called previously.

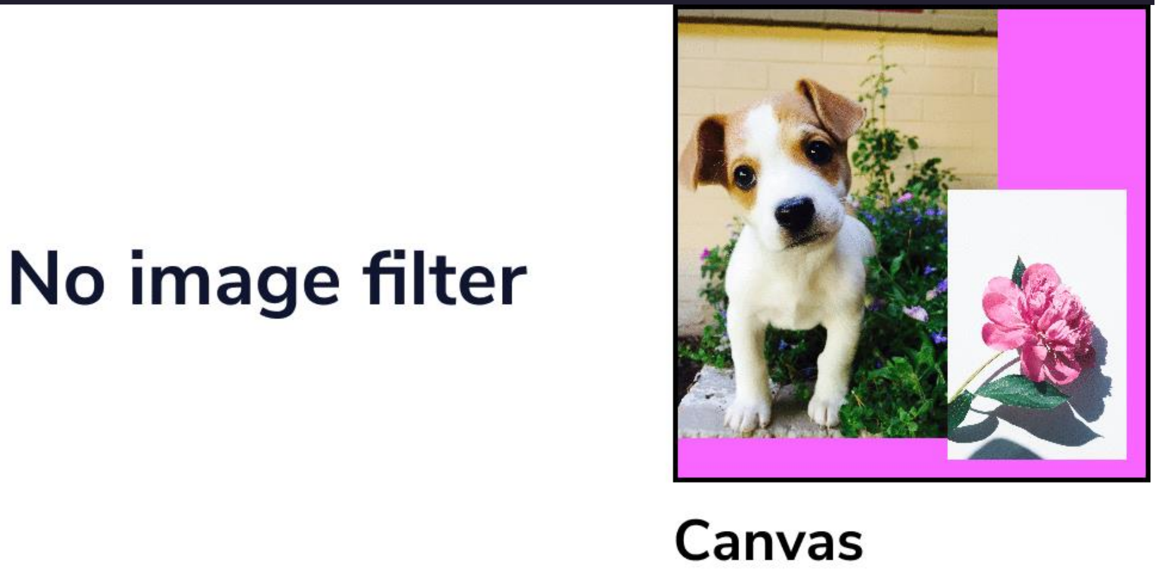


It also lets us apply filters to certain parts of the canvas—for example, we can apply a filter to the entire canvas, then draw new elements that’ll be unaffected.



To apply filters across an individual p5 image element, use the `.filter()` method. You'll need to call this before drawing the image.

```
//Invert colors in an image
img.filter(INVERT);
image(img, 0, 0);
```



We can't, however, apply filters to individual video elements (though you could instead draw a video to the canvas, then add a filter to the entire canvas).

A weakness of the `filter()` function is its slow performance—especially when called frequently. Later, we'll learn how to achieve similar effects (and more!) with pixel manipulation.

Instructions

1.

In this exercise, you'll see we've already drawn four identical images of a cute puppy to the screen. Let's turn this sketch into pop art—or rather, pup art!

In the `setup()` function, add a `GRAY` filter to `img1` before it is drawn to the canvas.
Hint
Remember the syntax for applying a filter to an individual image element is:

```
imageElement.filter(TYPE);
```

The `.filter()` method must be called before the image element is drawn to the canvas.

2.

Before `img2` is drawn, apply an `INVERT` filter to it.
Hint
To apply an invert filter, the `TYPE` should be set to `INVERT`.

3.

Before `img3` is drawn, apply a `POSTERIZE` filter to it. For the additional parameter, use the value `4`.
Hint
Remember that some filter types, like `POSTERIZE`, require one additional numerical argument.

4.

Before `img4` is drawn, apply a `THRESHOLD` function to it. Look up details on how to apply it in the p5.js reference.
Hint
`THRESHOLD` is a type of filter that optionally takes in an additional numerical argument between `0` and `1`. If no additional argument is provided, p5.js will use the parameter `0.5`.